Data Search Method for HDD

Hideki Kitao
Masahiko Higashiyama
Shinji Fukuda

Abstract

Fujitsu Ten has recently developed data search software that is targeted at audio title searching and is also suitable for handling any text data in list format.

The software is provided with functions for registering, editing, deleting and searching, and for acquiring the search results. It compiles each of various "groups" of data into a separate index containing the "record" (line) positions of the data items, thus enabling construction of a flexible system permitting search objects and output objects (results) to be selected separately. Further, the index component is loaded into the memory, realizing high-speed and memory-saving searches requiring fewer accesses to the disk.

This paper presents an overview of the newly-developed data search software plus its basic performance.

This software is included in the HDD-AVN launched in November 2002, forming part of its "Music Juke" functions.

1 Introduction

Every year sees more and more people enjoying music and moving visuals via PCs and the Internet. Alongside that there is an ongoing rise in the popularity of in-vehicle equipment permitting enjoyment of music data loaded at home. Such equipment is fitted with a CD drive or HDD (hard disk drive) able to play back MP3 data and thus can handle several hundreds to several thousands of music tracks.

At the same time in-vehicle equipment with functions for connection to the Internet via a mobile phone is on the increase, and it is anticipated that this will lead to an expansion in the data stored in the HDDs to include POIs (points of interest) and e-mail in addition to the map and music data they have handled hitherto.

As the amounts and varieties of the data increase, handling of the data becomes troublesome and breeds in users a desire to "get quickly to the information they need". Given this situation, search functions assume an important role.

We have recently developed data search software that is targeted at audio title searching and is also suitable for handling any text data in list format. Below we present an overview of this software.

This software is included in the HDD-AVN launched in November 2002, forming part of its "Music-Juke" functions.

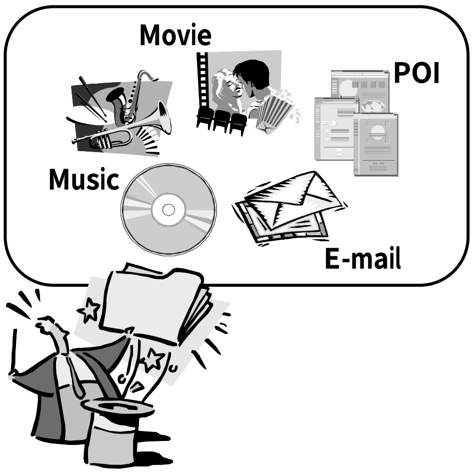


Fig.1 Rapid searching of highly varied data

2 Aims of the development

Our recent development work has realized functions for searching compressed music data loaded into HDDs carried by in-vehicle equipment. The data search software for this purpose performs mediation between multimedia data and the application using it, by means of the user database. The user database and multimedia data are accessed using a file system (refer to Fig. 2).

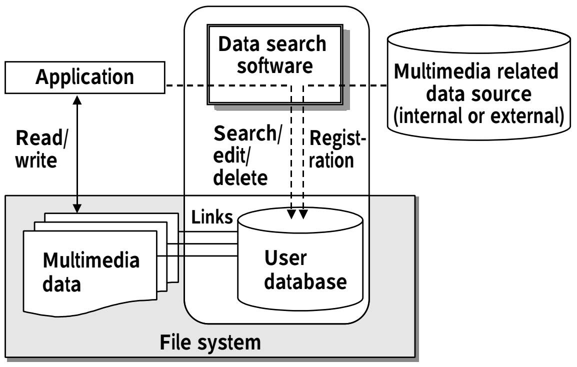


Fig.2 Data search software's place in the setup

2.1 Requirements for the data search software

①Must be high-speed

It is an essential condition for in-vehicle equipment that it can be operated in a brief space of time while the vehicle is being driven or is halted at traffic lights, etc. The computing power of CPUs for in-vehicle equipment is however less than 1/10 that of CPUs for PCs, and because of this the amount of computational processing must be kept as small as possible. Though HDDs tend to be thought of as high-speed devices, in fact they are poorly suited for random accessing of detailed data since they are connected to an external I/F and have the property of generating overhead due to ATA, SCSI or similar protocols when data is written into them (refer to Table 1). Accordingly one requirement was to make innovations that would reduce to as low a level as possible the number of random accesses to the HDD.

Table 1 Sample HDD accessing speeds (ATA PI0 mode 0)

Transmission method	Reading	Writing
64 KB×32 accesses	500KB/s	300KB/s
24 MB×1 access	2.2MB/s	1.9MB/s

②Must operate using a small amount of memory

Generally the total memory carried by medium-class in-vehicle equipment is just several MB, and even in high-class equipment is no more than 32 MB. This means that individual functions such as search software are required to operate on no more than 1 MB of memory at the most.

③Must be able to assign multiple pieces of information to a single stored data item

Whatever the kind of information, there will be several pieces of information present within each stored item of data. When the system handles audio titles for instance it will be essential for each data item to contain the track name, the artist's name, the album name, and the storage location of the data. Hereafter, a data item containing such a packet of multiple pieces of information is termed a "record".

④Must be provided with functions for storing and editing information

As adequate information can rarely be obtained from the data that the software stores, information obtained from separate data sources has to be saved somewhere else - in a location separate from the such sources (in Fig. 2 it is saved in the user database) - for utilization by the software. Accordingly the software must have functions built into it for registering, editing, deleting and searching such information.

⑤Must keep the "output objects" (search results) separate from the search objects

Suppose the software receives the instruction "search for tracks by artist "○○○". The objects that are searched (hereafter referred to as a "group") will be "artist names" and the application will request the search software to find data records with the name "○○○" from this group. However the data (output objects) that the application submits to the user will be a list consisting of the track titles contained in the records that came up as hits in the search. This submitted data may on another occasion be turned into a list of album names, and so should be handled as separate items from the search objects.

Table 2 gives the target specifications for the newly developed data search software. In consideration of the HDD-AVN specifications obtaining at the outset of the development, SH7709A was adopted for the CPU and micro Itron for the OS.

Table 2 Target specifications for data search software

Item	Target spec.
Search speed	Less than 1 second from search key input to results output
Number of search objects	1500
Check method	Full match
Functions	Registration, editing, deleting, searching, acquisition of search results
Operating memory	No more than 500 KB

3 Overview of the data search software

Fig. 3 shows the overall configuration of the search engine that carries the newly developed software. The software creates a table in the database so as to manage the data, and prepares data indexes that it uses for executing searches.

3.1 DBMS (database management system)

1) DBMS (database management system)

The DBMS is the interface between the application and the database. It receives requests for operations on the data (registration / editing / deletion / searching), calls up the various functions for table management, index management and record management, and executes operations regarding the database.

2) Table management component

This performs management of the information that makes up the data items in the table constructed in the database and in the indexes prepared for use in searches. Additionally it responds to parameter queries from the index management and record management components.

3) Index management component

This performs management of the data indexes created for rapid searching of the information registered in the table, and communicates storage locations of data in response to search requests.

4) Record management component

This manages the data in the table in the database, for which purpose it regards each line in the table as 1 packet of data (as will be described later). Treating each such packet as 1 "record", this component registers data received from the application into the records and extracts data from the records.

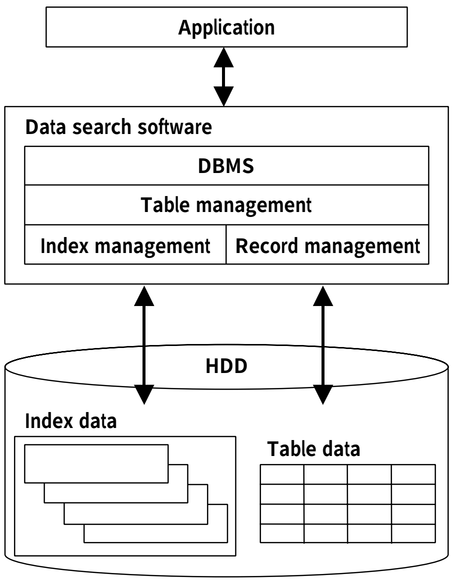


Fig.3 Configuration of data search engine

3.2 Table structure

①Table data

The registered data is stored in a 2-dimensional table. Structurally the data in the table can be expressed in terms of "records" (lines) and "groups" (columns). The table is constructed in the database and used for management of data. For the various data items stored in the table there are defined Record IDs and Group IDs that express the record and group locations respectively, and the application uses these IDs to access the data (refer to Fig. 4).

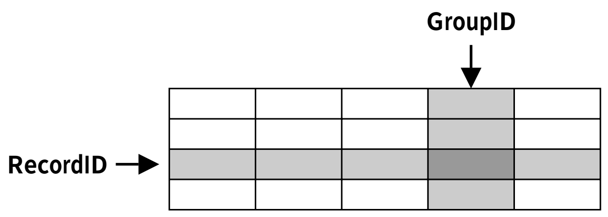


Fig.4 Accessing of table data

②Table customization

To realize versatility that will enable use with many different applications, provision must be made for the fact that the numbers and types of the data objects that are registered and searched will differ with each application. From each application, the present data search software acquires the data that will define the table requires - number of records, number of groups, and types of data to be registered (character strings, numeric values) - then uses such data to create a customized table within the database (refer to Fig. 5).

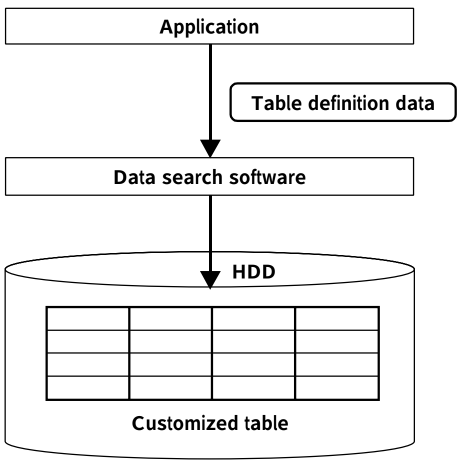


Fig.5 Preparation of table data

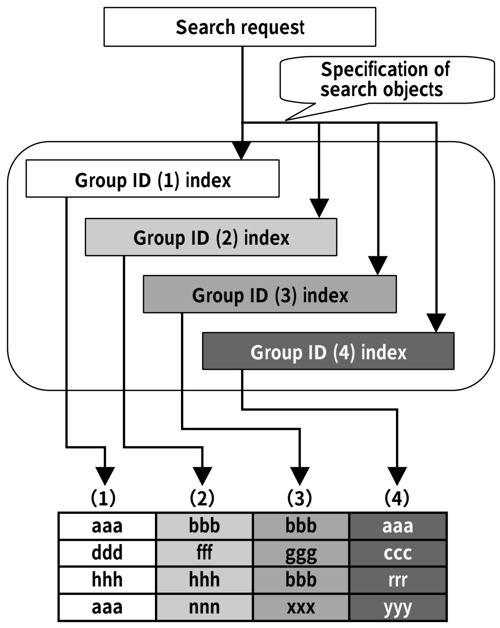
3.3 Index structure

①Index data

So as to execute search processing rapidly, the data search software compiles from the table data indexes of data that will facilitate searching. The data in these indexes are such as resemble items in a table of contents, and state the record IDs of the corresponding registered data. To find the records in the table that contain a target data item, it is simpler to consult the indexes to determine which records contain the item than to scan through all of the data right from the top of the table.

②Index scope

Indexes are compiled for each group of data in the table constructed in the database, so that each group can be searched individually. When defining the table the application categorizes the groups according to the attributes, etc., of the registered data, thus permitting searches of a specified object range such as artist name searches or album name searches (refer to Fig. 6).



(1)	(2)	(3)	(4)
aaa	bbb	bbb	aaa
ddd	fff	ggg	ccc
hhh	hhh	bbb	rrr
aaa	nnn	xxx	yyy

Fig.6 Scopes of indexes

③High-speed and memory-saving operation

Using indexes permits execution of high-speed searches, but only if the indexes are stored in the memory. One cannot expect high-speed search processing if the indexes are read out from the HDD, which is far slower to access than the memory. Thus it is important to keep the index data down to a quantity sufficiently small that the indexes can be permanently deployed in the small-capacity memory carried by in-vehicle equipment. The present data search software achieves this by constructing the indexes from Record IDs, thus compressing the volume of the index data.

④Index structure

This data search software uses the hash method for the index structure. The hash method yields a data structure that permits execution of registration / deletion / search operations without any dependence on the number of registered data items.

The principle of such structure is to link the data directly to the positions in which they are stored, through the introduction of a function h(x) that maps a key value x onto subscripts in the data array. Ordinarily the hash method does not permit registration of identical data, and the number of search matches it yields is 1.

In the present data search software however the method is expanded so as to permit identical data to be registered. This means that where identical data is stored in multiple records, a search for such data will return all of the records that match it.

⑤Search method

The present data search software is able to receive a search key list and execute multiple searches simultaneously on the basis of the list. The software holds lists of the record IDs of the results of each of the searches in a buffer, and when the searching has ended it extracts and merges the duplicated IDs from all of the record ID lists so as to execute a narrowed-down search (AND search). Alternatively, an expanded search (OR search) can be executed, by switching the post-search duplicated ID extraction processing to the OR routine, which omits the duplicates and extracts and merges all of the IDs that matched the search (refer to Fig. 7).

A list of the record IDs that matched is returned to the application as the results of the search processing. The application can use this record ID list to access the table so as to acquire the data stored under the search result IDs. By switching the data it requires according to particular purposes, the application is able to process such acquired data with the consumption of only a small amount of memory. For instance if it has to switch from a display of track titles matching a search to a display of the recording dates, it can do so without implementing another search, since it will be able to make the buffer containing the title data identical with the buffer containing the recording date data. In this way it can perform processing in a memory-saving manner.

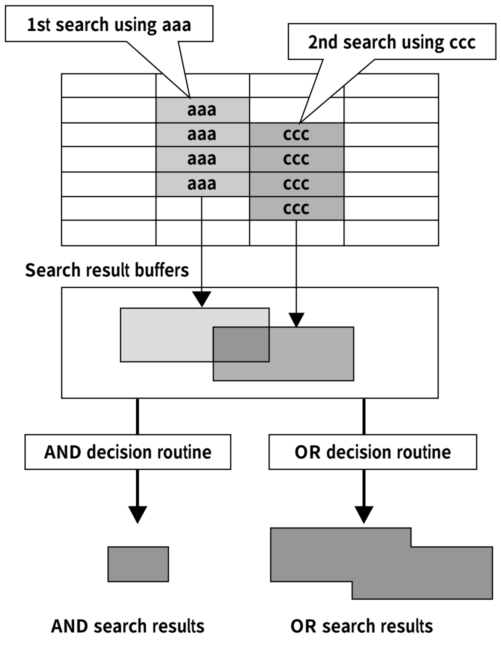


Fig.7 Search method

3.4 Reason for index structure selected

The present data search software employs the hash method for its index structure. Another method in frequent and general use for such structure is the well-known "B-tree" method, employing a "tree"-like structure.

The hash method uses a simple data structure consisting of a data array constituting a hash "table", plus a linear list for managing data with identical hash values. Registering / deleting of data to / from the hash structure can be executed with high speed since it merely requires operations on the linear list. By contrast, the B-tree method configures a balanced multiple-branching tree structure composed of "roots" that are the start points, "branching points" that are the nodes, and "leaves" that are the terminal extremities. However this method entails a complex data structure because its nodes have multiple junctions, so that registration / deletion necessitates complex processing involving large amounts of calculations in order to maintain the balance of the structure.

Because it requires multiple junction data for each of its nodes, the B-tree method consumes large amounts of memory and must sequentially read in a part of its node data from an HDD in order to maintain its tree structure.

By contrast the hash method consumes only a small amount of memory thanks to its simple data structure, which means that it can maintain a loading state in memory with respect to the indexes produced. Even with frequent registrations, deletions and changes, the hash method can handle the processing via the memory, enabling high-speed processing with few accesses to the HDD.

Table 3 Characteristics of hash and B-tree methods

Item	Hash	B-tree
Structure	○	×
Memory consumption	○	×
Search speed	○	○
Registration/Deleting speed	○	×
Change speed	○	×
Selection	◎	—

Further, the hash method is suited to small-to-medium scale systems requiring registration, deletion and change functions. For the various reasons given above, the hash method is the one adopted for the index structure in the present data search software.

4 Performance measurement

To execute high-speed searches the present data search software is equipped with functions for preparing indexes of table data and finding search-matching data in them. We verified the efficacy of these indexes in search processing by measuring the software's search speed.

4.1 Configuration of measuring system

Fig. 8 shows the configuration of the measuring system.

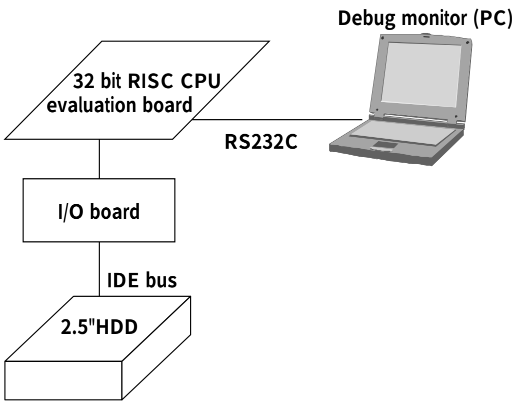


Fig.8 Configuration of performance measuring system

4.2 Measurement method

To verify the efficacy of the indexes, we separately prepared a set of software that employs a search method that does not include any management data, and used it for comparison with the measurement results. This software without management data performs search processing by sequentially reading out table records from an HDD and comparing them. The performances of the present data search software and of the software for comparison were measured according to the procedure below.

①Preparation of registered data for use in measurement

To measure the performance it was necessary to register data in the table constructed in the database. The following 3 types of data were prepared for this purpose, so as to measure the maximum search durations. The group prescriptions for such data are given in Table 4 and the record structure in Fig. 9.

1) Type A

These were numerical values 1 through 1500, which were loaded into the records of the top group and tail-end group.

Examples: 0001, 0002, 0003, ...

2) Type B

These were identical character strings of the maximum size containable in one record, which were loaded into all of the records of arbitrarily selected groups.

Examples: "A E I O U", "A E I O U" .....

3) Type C

These were differing character strings of the maximum size containable in one record space, which were loaded into all of the record spaces of arbitrarily selected groups.

Examples: "A0001", "A0002", "A0003" .....

Table 4 Group prescriptions

Group	Data	Type
1	Numeric values (2)	Type A
2	Identical character strings (256)	Type B
3	Unique character strings (256)	Type C
4	Identical character strings (256)	Type B
5	Unique character strings (256)	Type C
6	Identical character strings (256)	Type B
7	Unique character strings (256)	Type C
8	Identical character strings (256)	Type B
9	Unique character strings (256)	Type C
10	Numeric values (2)	Type A

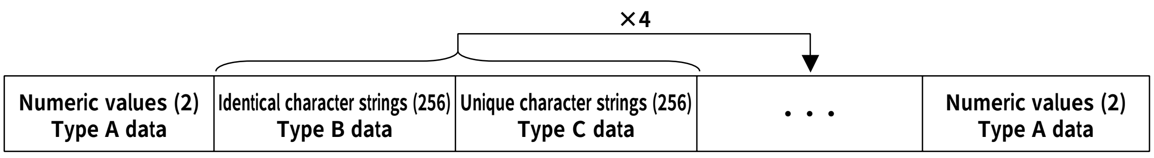
* Number of bytes in parentheses

The database size and memory consumption when the above were used as the registered data are given in Table 5.

Table 5 Database size and memory consumption

	Data search software	Software for comparison
Indexes	156	0
Table	3012	3012
Memory consumption	284	128

* Units: kilobytes


×4
Numeric values (2) Type A data
Identical character strings (256) Type B data
Unique character strings (256) Type C data
・・・
Numeric values (2) Type A data

Number of bytes in parentheses

Fig.9 Structure of registered data "records"

②Measurement of performance

The processing durations for the following items were measured.

1) Registration

Data to fill 1500 records were registered into the table at once as a single batch, and the registration duration per record was measured.

2) Search I

The times taken to search for and find the type A data loaded into the first and last records of the top group and of the tail-end group were measured.

3) Search II

For each group of B type data, a search for B type data was executed and the time taken until a result was returned was measured.

4) Search III

For each group of B type data, a search for non B type data was executed and the time taken until "Not found" was displayed was measured.

5) Search IV

For each group of C type data, searches for the data contained in the first record, last record and 10 other randomly selected records were executed and the duration of each search was measured.

6) Deletion

The 1500 records were all deleted at once, and the delete duration per record was measured.

4.3 Results of performance measurement

The results of the performance measurement are given in Table 6, from which it can be seen that the present data search software is some 100 times faster than the software used for comparison. Analysis of the results shows that with both of the programs the time taken up by disk accessing accounts for much the greater part of the total processing time.

Disk accessing is drastically reduced with the present data search software because it employs a method of first running searches using the indexes, then accessing the table and checking whether the search results are correct. This fact can be credited with the increased speed achieved by the new software. Although there has been much research hitherto into search methods that would cut down calculation volume, presumably searches using less information should be effective to reduce calculations for searching of data recorded into HDDs. We hope to pursue further research focusing on this issue in the future.

In the newly developed software, the following have been realized:

- Functions for registering, editing, deleting and searching, and for acquisition of search results
- Memory consumption of 284 Kbytes for 1500 items of registered data
- Speed of 10 msec per search on average

Thus the target specifications were achieved.

5 Application to HDD-AVN title data management

To enable application of the present data search software to HDD-AVN title data management, we developed the software by adding title play list functions to its search engine, and expanded the number of search targets to 300 songs. Fig. 10 in the following item shows the configuration of HDD-AVN title data management; below we present the main functions of HDD-AVN title data management.

1) Play list creation functions

Can create various play lists including album play lists and artist play lists.

2) Play list sequencing functions

Can rearrange play list item sequences by moving titles that are selected from a list to the top of the list. Can also alter the sequence of the play lists themselves.

Table 6 Results of performance measurement

	Data search software				Software for comparison			
	Total processing time		Disk accessing time		Total processing time		Disk accessing time	
	Average	Max	Average	Max	Average	Max	Average	Max
Registration	30	—	20 (6)	—	370	—	340 (752)	—
Search I	10	10	10 (1)	10 (1)	780	1060	750 (1500)	1020 (1500)
Search II	10	10	10 (1)	10 (1)	1420	1660	1070 (1500)	1340 (1500)
Search III	10	10	10 (0)	10 (0)	1370	1660	1080 (1500)	1350 (1500)
Search IV	10	40	10 (2)	30 (2)	1160	1530	990 (1500)	1330 (1500)
Deleting	10	—	10 (1)	—	10	—	10 (1)	—

* Units msec, minimum measurement unit 10 msec.
* No Max. values are given for registration and deleting because measurement was of registration/erasure of all 1500 data items at once.
* Figures in parentheses after disk accessing times are the number of accesses.

3) Sequencing cancellation functions

Can sort out lists that have had their item sequences rearranged, and return them to the default sequences.

4) Play list creation functions

Can create individualized play lists customized by the user.

5) Title erasure functions

Can erase from the play lists particular titles that the user specifies.

6) Track information acquisition functions

Can acquire track-specific information on titles that are selected from play lists.

7) Random acquisition functions

Can randomly acquire track information registered in the play lists.

8) Track information change functions

Can change the titles of registered tracks or the titles of the play lists, etc.

9) Registered data searching functions

Can search the track numbers of tracks already registered. Can prevent double registration.

6 Conclusion

The newly developed data search software is targeted at title lists in audio and similar equipment, but is also capable of being applied in other applications as a piece of basic search software.

To expand the software's search repertoire to include e-mail and stored Web content, etc., we will be pressing ahead in the future to add functions for high-speed searching of keywords in text and to create mechanisms enabling various different media to be handled by the same software. In this way we will hope to achieve software of even greater versatility.

Reference literature

1) Makoto Takizawa, RDBMS Gijutsu Kaisetsu ("Technical Explication of RDBMS"), Software Research Center, Inc., 1996

2) Robert Sedgewick, Algorithms, 2nd ed., Addison-Wesley Publishing Company, Inc., 1988

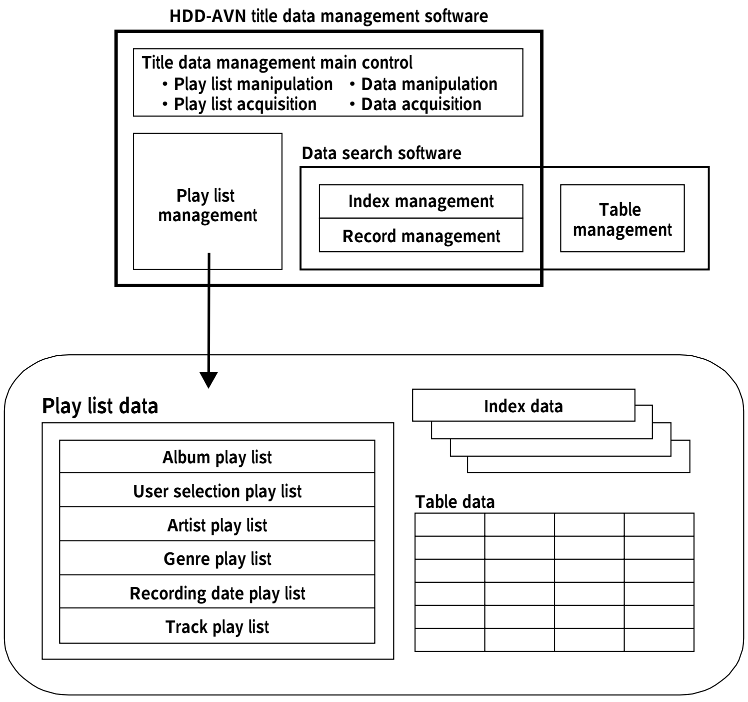


Fig.10 Configuration of HDD-AVN title data management

Profiles of Writers

Hideki Kitao

Entered the company in 1992. Since then, has been involved in the software development for digital application devices, such as noise suppression and voice processing.
Currently in the Software R&D Department of the System R & D Division, A.V.C. Products Group.

Masahiko Higashiyama

Entered the company in 2001. Since then, has been involved in the development of data search middleware.
Currently in the Software R&D Department of the System R&D Division, A.V.C. Products Group.

Shinji Fukuda

Entered the company in 1979. Since then, has been involved in the development of vehicle mounted information devices.
Currently the Manager of the Software R&D Department of the System R&D Division, A.V.C. Products Group.